Explainable Graph Classification

With the advent of network data, there has been plenty of interest and applications in developing high-quality machine learning (ML) techniques for classifying graph objects, including cheminformatics (e.g., compounds that are active or inactive for some target) and bioinformatics (e.g., classifying proteins into different families), malware detection and classification with call graphs, telecommunication networks (e.g., classifying customers and anomalies based on their calling behavior and churn prediction), power grids, internet-of-things, and wearable devices (e.g., detecting anomalies), trajectories, online check-ins, and social networks (e.g., classifying users based on their locations and feeds on Twitter, Facebook, etc.). Unfortunately, the space of graphs contains almost no mathematical structure, and consequently, makes it difficult to build a classifier directly over the graph space. In the past, graph kernels (e.g., random walks, shortest paths, cyclic patterns, subtrees, graphlets, and subgraph kernels) [15], feature based classification (e.g., frequent subgraphs, significant subgraphs, label features) [13, 17, 18], graph embedding (e.g., subgraph2vec [11, 19]) and deep learning (e.g., DeepGraphs [21], GraphGAN [16], GCN [2, 3, 5, 7]) methods were developed. Due to the challenges of feature engineering, coupled with the hardness of subgraph mining and isomorphism testing, graph embedding and deep learning methods are increasingly becoming popular.

However, state-of-the-art machine learning and deep learning models are "black-box" in nature. Hence, an increasing number of researchers, practitioners, and policy makers are realizing that much needs to be done to promote transparency of ML models. Network data are complex, noisy, massive-volume, and content-rich. End users (e.g., data scientists, business analysts, engineers, and developers) often find them difficult to query and explore. If querying and exploration of these datasets are difficult, will not interpreting the results of (black-box) machine learning tools over graphs be even harder? Hence, it is critical to provide human-understandable explanation and support answering "why" and "why-not" questions over prediction results on such datasets.

In this project, we shall build our ML classifiers on top of learned and explainable features, e.g., association and temporal rules, correlated patterns, so to explain the outcomes of the machine learning models, thereby adding transparency to our designed algorithms. A brief description of our methodologies are given below.

(1) Graph Features Embedding and Graph to Image Conversion: To train a powerful classifier on top of identified graph features (e.g., frequent subgraph patterns [18]), we devise a novel method to convert graphs into $2D$ image grids by embedding its frequent subgraph features and motifs in a lower dimensional vector space. Notice that unlike [2, 3, 5, 7], we do not consider GCNs over the graph Laplacian matrix due to lack of transparency. Furthermore, while embedding subgraph features and motifs, we consider their correlation as follows. In the context of words and documents, the skip-gram model [9] maximizes the co-occurrence probability among the words that appear within a given context. Analogously, given a dataset of graphs, we consider the neighborhood of a subgraph as its context. Subsequently, following the language model

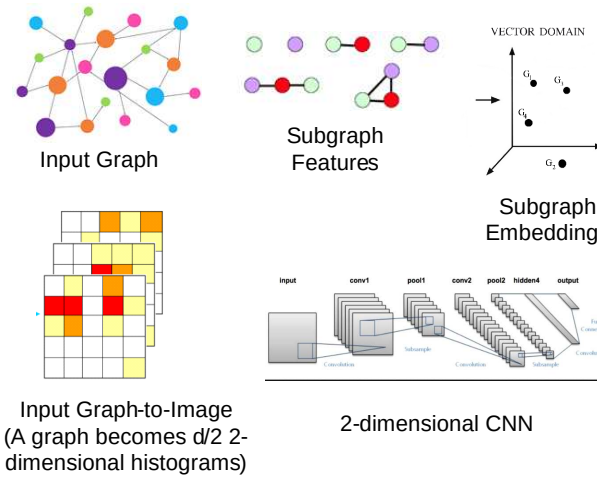

Figure 1: Our approach to convert graphs as images via subgraph features embedding. Given a set of input graphs and their subgraph features, we embed each subgraphs in a d-dimensional plane, following the Skipgram model. Next, each input graph is represented by $d/2$ 2-dimensional histograms. The first histogram is computed from the first two dimensions, the second histogram from dimensions 3 and 4, and so on. Bins of a histogram can be viewed as pixels, where each pixel is associated with a vector of size $d/2$, whose entries are the presence/counts of a subgraph feature falling into that bin in the corresponding 2-dimensional slice of the embedding space. Finally, each graph represented as $d/2$ 2-dimensional histograms is handled by $2D$ CNNs.

training process with the subgraphs and their contexts, we learn the intended subgraph embedding [11, 19]. Clearly, similar subgraphs will have similar representations in the embedding space.

Next, as depicted in Figure 1, we represent a graph with $d/2$ 2-dimensional histograms, where d is the embedding dimensionality of subgraph features. The first histogram is computed from the first two dimensions, the second histogram from dimensions 3 and 4, and so on. Using computer vision vocabulary, bins of a histogram can be viewed as pixels, and the 2-dimensional slices of the embedding space as channels. In our case, instead of having 3 channels (R,G,B) as in color images, we have $d/2$ of them. That is, each pixel (i.e., each bin) is associated with a vector of size $d/2$, whose entries are the presence/counts of a subgraph feature falling into that bin in the corresponding 2-dimensional slice of the embedding space. Thus, we introduce an innovative way to represent graphs as image-like structures that allow them to be handled by state-of-the-art $2D$ CNNs for supervised classification [14].

(2) Learning Important Features via Relevance Propagation: While the recent development in deep neural networks and their accuracy results are impressive, the understanding of what makes a neural network arrive at a particular decision is still an open problem. A typical way of comparing features extracted by a first layer of a deep network is by looking at "filters" learned by the model, e.g., stroke detectors on digit data, edge detectors on images, etc. [6, 12]. Visualization and detection of higher-level features have been discussed in [4, 10, 20]. In contrast, layer-wise relevance propagation (LRP) [1] operates by building for each neuron of a deep network a local redistribution rule, and applying these rules in a backward pass in order to produce pixel-wise

decomposition, thereby interpreting which pixels of an image are most important for the prediction of an image. Both methods have their drawbacks. While the first approach (i.e., visualizing higher-level features) does not relate these higher-level features to the predictions that they cause; the second approach (i.e., layer-wise relevance propagation) relate instance-specific features to instance-specific predictions, and the explanations that they produce do not generalize beyond a single instance. Recently, influence-directed explanation for Deep CNNs [8] has been proposed that instead combines the benefits of both techniques: It identifies neurons with high influence on the models behavior, and then uses existing techniques (e.g., visualization) to provide an interpretation for the concepts that they represent.

In this project, with the help of layer-wise relevance propagation (LRP) [1] and influence-directed explanation for Deep CNNs [8] methods, we shall identify the neurons with high influence on the model's behavior, and then by using visualization techniques, we shall provide an interpretation for the concepts that they represent. Since our previous step, i.e., graph-to-image conversion is based on subgraph features and motifs, we aim at identifying the important features with such backward relevance propagation techniques. Furthermore, we shall compare the most significant features learnt via our framework with a more direct supervised approach, e.g., LEAP search and significant subgraphs [13, 17].

Contact: Arijit Khan, Assistant Professor, School of Computer Science and Engineering, Nanyang Technological University, Singapore; (arijit.khan@ntu.edu.sg).

References

[1] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Mller, and W. Samek. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE*, 10(7):0130140, 2015.

[2] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral Networks and Locally Connected Networks on Graphs. In *ICLR*, 2013.

[3] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NIPS*, 2016.

[4] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing Higher-Layer Features of a Deep Network. Technical Report 1341, Universite de Montreal, 2019.

[5] M. Henaff, J. Bruna, and Y. LeCun. Deep Convolutional Networks on Graph-Structured Data. In *CoRR abs/1506.05163*, 2015.

[6] G. E. Hinton, S. Osindero, M. Welling, and Y. W. The. Unsupervised Discovery of Nonlinear Structure Using Contrastive Backpropagation. *Cognitive Science*, 30(4):725–731, 2006.

[7] T. N. Kipf and M. Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, 2017.

[8] K. Leino, L. Li, S. Sen, A. Datta, and M. Fredrikson. Influence-Directed Explanations for Deep Convolutional Networks. In *CoRR abs/1802.03788*, 2018.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. In *CoRR abs/1301.3781*, 2013.

[10] G. Montavon, M. L. Braun, and K.-R. Mller. Kernel Analysis of Deep Networks. *Journal of Machine Learning Research*, 12:2563–2581, 2011.

[11] A. Narayanan, M. Chandramohan, L. Chen, Y. Liu, and S. Saminathan. subgraph2vec: Learning Distributed Representations of Rooted Sub-graphs from Large Graphs. In *CoRR abs/1606.08928*, 2016.

[12] S. Osindero and G. E. Hinton. Modeling Image Patches with a Directed Hierarchy of Markov Random Fields. In *NIPS*, 2007.

[13] S. Ranu and A. K. Singh. GraphSig: A Scalable Approach to Mining Significant Subgraphs in Large Graph Databases. In *ICDE*, 2009.

[14] A. J.-P. Tixier, G. Nikolentzos, P. Meladianos, and M. Vazirgiannis. Graph Classification with 2D Convolutional Neural Networks. In *CoRR abs/1708.02218*, 2017.

[15] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt. Graph Kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

[16] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo. GraphGAN: Graph Representation Learning with Generative Adversarial Nets. In *CoRR abs/1711.08267*, 2017.

[17] X. Yan, H. Cheng, J. Han, and P. S. Yu. Mining Significant Graph Patterns by Leap Search. In *SIGMOD*, 2008.

[18] X. Yan and J. Han. gSpan: Graph-Based Substructure Pattern Mining. In *ICDM*, 2002.

[19] P. Yanardag and S. Vishwanathan. Deep Graph Kernels. In *KDD*, 2015.

[20] M. D. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. In *ECCV*, 2014.

[21] M. Zhang, Z. Cui, M. Neumann, and Y. Chen. An End-to-End Deep Learning Architecture for Graph Classification. In *AAAI*, 2018.